Department : **CSE-DS**

Year & Semester : **III YEAR V SEM**

Sub Code & Sub Name : **23CSE243T && SOFTWARE ENGINEERING**

Unit-I

S.No	Part-A Questions
1.	Define software engineering.
2.	Define abstraction in software engineering?
3.	What is decomposition in the context of software design?
4.	List out the phases of Software life development life cycle?
5.	Differentiate between abstraction and decomposition with an example.
6.	State the main advantage of the Iterative Waterfall model over the classical Waterfall model.
7.	List two key milestones in the evolution of software engineering techniques.
8.	Name two Agile models and their core principles.
9.	What is project planning in software project management? What are the essential activities in project planning?
10.	Define project estimation and mention one technique used for it.
11.	What does COCOMO stand for, and what is its purpose?
12.	What are the 3 classes of software?
13.	Define team structure in software projects.
14.	What is risk management in software projects?
15.	What is UFP?
16.	What is PERT and Gantt charts?
17.	Explain the metric for project size estimation (Loc and fp)?

S.No	Part-B Questions
1.	Compare and contrast abstraction and decomposition in software engineering. Provide examples to illustrate how they contribute to software design.
2.	Discuss the evolution of software engineering techniques from the 1960s to the present day, highlighting major advancements and challenges.
3.	Explain the Iterative Waterfall model in detail, including its phases, advantages, and limitations compared to other SDLC models.
4.	Describe the Prototype model and the Evolutionary model. How do they handle requirements uncertainty
5.	Elaborate on the Spiral model with a neat diagram.
6.	Discuss Agile models in SDLC with a neat diagram.
7.	Explain about prototyping model. Discuss it's advantages and disadvantages?
8.	Explain about RAD model with a neat diagram.

9.	Explain about Agile model. Discuss it's advantages and disadvantages?
10.	Explain in detail the following project size estimation techniques. i) Lines of Code (LOC) ii) Function Point (FP) Metric
11.	Analyze the concept of COCOMO model and its types(Basic, Intermediate and complete) with example.
12.	Define Risk Management and Explain types of risk management.
13.	Explain software project management in detail
14.	Discuss project scheduling
15.	Explain risk management and configuration management in software projects. How do they ensure project success
16.	Compare the following project size estimation techniques. i) Expert judgement ii) Delphi technique

Unit-II

S.No	Part-A Questions
1.	Define software and software engineering
2.	What do you mean by myth? How many types of software myths are there.
3.	Name the characteristics of a Good SRS Document.
4.	What is meant by Traceability in SRS
5.	What do you mean by good SRS
6.	How do you represent complex requirements?
7.	What are decision tables and decision trees?
8.	What is meant by Formal system? Name any 2 techniques?
9.	What is decision tree? Give one example?
10.	Explain axiomatic specification.
11.	Explain about algebraic specification.
12.	Explain the characteristics of software.
13.	What are functional and non functional requirements?
14.	What are the attributes of Bad SRS?
15.	Who are the users of SRS documentation
16.	What are the types of software?
17.	What is 4GL'S?

S.No	Part-B Questions
1.	Explain the process of requirement gathering and analysis? And also explain ways of doing requirement gathering and analysis?
2.	What are software myths? Explain in detail.
3.	What is SRS? What are the characteristics of good SRS and bad SRS?
4.	Explain IEEE 830 guidelines for customer requirements organizing.
5.	Contrast the following categories of requirements.

	I) Functional requirements ii) Non-functional requirements
6.	A) What is a Formal Technique? Identify important concepts in formal methods, and examine the merits and demerits of using formal techniques. b) What is Axiomatic Specification? How to develop an axiomatic specification with an example
7.	How to represent complex requirements (Decision tree and decision table)? Explain using diagrams?
8.	What are the advantages of SRS document?
9.	Explain merits and demerits of Formal requirement specification?
10.	A) Explain axiomatic specification. B)Explain about algebraic specification
11.	A) Explain the advantages of SRS? B) What are the problems without using SRS document?
12.	Name some important categories of users of the SRS document and their needs?
13.	What is the role of system analyst in requirement analysis?

Unit-III

S.No	Part-A Questions
1.	What are the characteristics of good software design.
2.	What is coupling and cohesion?
3.	Explain the concept of control hierarchy in software design.
4.	What is control abstraction, and how does it simplify software design?
5.	What is structured analysis?
6.	Define fan-out and fan-in in software design.
7.	What is the key difference between object-oriented and function-oriented design?
8.	What is meant by Generalization?
9.	What is meant by stereotypes? What the use of includes and extends?
10.	What is a Data Flow Diagram (DFD), and what is its primary purpose?
11.	What is meant by Association? Explain types of association?
12.	What is dependency relation? How to represent it.
13.	What are Aggregation and composition relationships?
14.	How to represent composition ,aggregation and multiplicity?
15.	List two types of UML diagrams used in software design.
16.	What is structured design, and how does it relate to detailed design?
17.	Mention two characteristics of a good user interface design.
18.	Differentiate between mode-based and mode-less interfaces with an example.
19.	What is component-based GUI development, and why is it used?

S.No	Part-B Questions
1	Discuss the principles of good software design and explain how they contribute to

	maintainable and scalable systems.
2	Explain cohesion and coupling in detail, with examples illustrating different types of cohesion and coupling.
3	Compare and contrast object-oriented design and function-oriented design, highlighting their strengths and weaknesses.
4	Provide a detailed overview of the SA/SD methodology, including its phases and applications in software development.
5	Explain structured analysis and how Data Flow Diagrams (DFDs) are used to model system processes.
6	Describe the conceptual model of UML and its role in object-oriented software design.
7	Explain the different types of UML diagrams and their specific uses in the software design process.
8	Analyze the characteristics of a good user interface and their importance in enhancing user experience.
9	Compare mode-based and mode-less interfaces, discussing their advantages, disadvantages, and suitable applications.
10	Explain the GUI design methodology, including the steps involved in component-based GUI development and its benefits.
11	Describe control hierarchy in software design, focusing on layering and its impact on system modularity.
12	Elaborate on control abstraction, including its role in managing complexity in large software systems.
13	Analyze the concepts of depth, width, fan-out, and fan-in in control hierarchy, and explain their significance in software design.

Unit-IV

S.No	Part-A Questions
1.	Define coding standards and give one example of a coding standard.
2.	What is the purpose of code review in software development?
3.	Name two types of software documentation used in the coding phase.
4.	Differentiate between black box testing and white box testing.
5.	What is debugging, and why is it important in software development?
6.	Define integration testing and mention one approach to it.
7.	What is the role of program analysis tools in software testing?
8.	Explain the concept of system testing in brief.
9.	What is performance testing, and why is it conducted?
10.	Define regression testing and its significance in software maintenance.
11.	Name two challenges in testing object-oriented programs.
12.	What is a coding guideline, and how does it differ from a coding standard?
13.	Mention one benefit of conducting a code review process.
14.	What is the purpose of black box testing in software validation?
15.	List two common debugging techniques used by developers.

S.No	Part-B Questions
1.	Discuss the importance of coding standards and guidelines, and explain how they improve software quality.
2.	Elaborate on the code review process, including its steps and benefits in ensuring robust software development.
3.	Explain the role of software documentation in the coding and testing phases, with examples of different types of documentation.
4.	Compare and contrast black box testing and white box testing, providing scenarios where each is most effective.
5.	Analyze the importance of performance testing in software development, and describe different types of performance tests.
6.	Discuss regression testing, its process, and its role in maintaining software reliability during updates.
7.	Describe the debugging process in detail, including common techniques and tools used to identify and fix defects.
8.	Discuss integration testing, its approaches (top-down, bottom-up, and hybrid), and their applications in software development.
9.	Explain the role of program analysis tools in improving code quality and testing efficiency, with examples of specific tools.
10.	Provide a detailed overview of system testing, including its objectives, scope, and methods used.
11.	Explain the challenges in testing object-oriented programs and propose strategies to address them.
12.	Describe how coding standards and guidelines can be enforced in a development team, with examples of enforcement mechanisms.
13.	Discuss the process of testing object-oriented programs, focusing on specific techniques like inheritance and polymorphism testing.
14.	Explain the relationship between integration testing and system testing, and how they contribute to overall software quality.
15.	Analyze the role of software documentation in supporting testing activities, including its impact on debugging and maintenance.

Unit-V

S.No	Part-A Questions
1.	Define software reliability and mention one way to measure it.
2.	What is statistical testing in the context of software quality?
3.	Name two key principles of software quality management.
4.	What is the purpose of the ISO 9000 standard in software development?
5.	List two maturity levels in the SEI Capability Maturity Model (CMM).
6.	What is the Personal Software Process (PSP), and what is its goal?
7.	Define Six Sigma and its relevance to software quality.
8.	Name two software quality metrics used to assess product quality.
9.	What is CASE, and what does it stand for in software engineering?
10.	Mention one role of CASE tools in the software development life cycle.

11.	List two characteristics of software maintenance.
12.	Define software reverse engineering and its purpose.
13.	What is a software maintenance process model?
14.	Name one method used for estimating software maintenance costs.
15.	What is a key issue in implementing a software reuse program?

S.No	Part-B Questions
1.	Discuss the concept of software reliability and explain how it can be measured and improved in software projects.
2.	Explain statistical testing in detail, including its methodology and applications in ensuring software quality.
3.	Describe the principles of software quality management and their role in delivering high-quality software products.
4.	Elaborate on the ISO 9000 standard, its components, and its application in software development processes.
5.	Discuss the SEI Capability Maturity Model (CMM), including its five maturity levels and their significance in process improvement.
6.	Explain the Personal Software Process (PSP) and how it contributes to improving individual developer productivity and quality.
7.	Analyze the Six Sigma methodology and its application in achieving software quality and process optimization.
8.	Discuss various software quality metrics, their types, and how they are used to assess and improve software quality.
9.	Explain the scope of CASE tools and their role in supporting different phases of the software development life cycle.
10.	Describe the CASE environment and how it integrates tools to enhance software development efficiency.
11.	Analyze the characteristics of software maintenance and the challenges faced during the maintenance phase.
12.	Discuss software reverse engineering, its process, and its importance in maintaining legacy systems.
13.	Explain the software maintenance process model, including its stages and how it ensures effective maintenance.
14.	Describe methods for estimating software maintenance costs, including factors that influence these estimates.
15.	Discuss the basic issues in implementing a software reuse program and propose strategies for effective reuse at the organizational level.